A Fuzzy Approach for Interpretation and Application of Ubiquitous Data Stream Clustering

Osnat Horovitz, Shonali Krishnaswamy and Mohamed Medhat Gaber
School of Computer Science and Software Engineering, Monash University
osnat.horovitz@gmail.com,
{Shonali.Krishnaswamy, Mohamed.Medhat.Gaber}@infotech.monash.edu.au

Abstract

Ubiquitous Data Mining is the process of analysing data emanating from distributed and heterogeneous sources in the form of a continuous stream with mobile and/or embedded devices. Unsupervised learning is clearly beneficial for initial understanding of data streams, and consequently various clustering algorithms have been developed and applied in UDM systems for the purpose of mining data streams. However, unsupervised data mining techniques require human intervention for further understanding and analysis of the clustering results. This becomes an issue as UDM applications aim to support mobile and highly dynamic users/applications and there is a need for real-time decision making and interpretation of results. In this paper we present an approach to automate the annotation of results obtained from ubiquitous data stream clustering to facilitate interpreting and use of the results to enable real-time, mobile decision making.

1. Introduction

Ubiquitous Data Mining is the process of analysing data emanating from distributed and heterogeneous sources in the form of a continuous stream with mobile and/or embedded devices. The ever-increasing computational capacity of small devices presents an exciting new opportunity for intelligent data analysis in applications and scenarios where the data is continuously streamed to the device [19] [16] and where there are temporal and other constraints (e.g. real-time information needs of mobile users) that necessitate the paradigm of analysis "anytime, anywhere" [24, 16].

In the last two years, UDM has made rapid strides and there are now techniques that have been developed to accurately and efficiently analyse rapid data streams in resource-constrained devices that have limited memory and processing capabilities [19, 16, 13, 14]. UDM algorithms aim to perform typical analysis tasks in an uncharted operational context that is characterised by one or more of the following constraints: limited computational resources, mobility (of users, devices and data sources themselves) and data that is generated and sent in real-time in a stream format with little or no potential for persistent storage. Typical application scenarios of UDM include:

1. *Road Safety* - Performing real-time analysis of sensory data in moving vehicles for crash prevention is emerging as a key application for UDM in Intelligent Transportation Systems as demonstrated in [16, 17].
2. *Sensor Networks* - Continuously monitoring and analysing status information received from sensors deployed in the Great Barrier Reef in order to reduce satellite communication by sending aggregated information rather than raw data generated from a myriad of sensor nodes [6]. Another application for UDM is where learning algorithms are used in conjunction with sensor toolkits such as MOTES™ to monitor the health of pigs in farms [5]. There are two sub-classes of sensor network applications of UDM – those where the sensors are stationary (such as the pig farm scenario) and those where the sensors themselves are mobile (e.g. sensors used in space exploration [23] or bio-sensors that are used to monitor schools of fish or sensors placed on animals).
3. *Mobile Users* - There are several applications that target the use of UDM to support intelligent and real-time information delivery to mobile users. For example, *MobiMine* is a system that allows a mobile stockbroker to monitor a portfolio from streamed stock market data while travelling [19], a travelling salesperson performing customer profiling [10], and the emerging class of mobile healthcare workers who require analytical and predictive models to support diagnosis.

The preceding applications clearly indicate the usefulness and possibilities that the emergence of ubiquitous data mining provides in addressing a class of real-world problems that was hitherto infeasible. It must be noted that within the gamut of UDM applications, there are applications that focus on analysing and building models of the rapid data streams, applications that focus on applying predictive models in a rapidly changing context and applications that combine the two approaches. It is also evident that there is a continuum with respect the levels of *mobility* and *user interaction* in these applications.

Unsupervised learning is clearly beneficial for initial understanding of data streams, and consequently various clustering algorithms have been developed and applied in UDM systems [6] [19] [16] for the purpose of mining data streams. However, unsupervised data mining techniques require human intervention for further understanding and analysis of the clustering results. This becomes an issue as UDM applications aim to support mobile and highly dynamic users/applications and there is a need for real-time decision making and interpretation of results. One option is, therefore, to perform exploratory unsupervised learning off-line, subsequently develop classificatory models and then apply them to ubiquitous data streams. In this paper, we propose an approach wherein we aim to develop an intermediary approach to transition from unsupervised clustering of ubiquitous data streams to interpreting and using the results in an online and continuous manner in a mobile device/environment. We extend our online and resource-aware ubiquitous data stream clustering algorithm – *Light-Weight Clusteing (LWC)* [12][15] with support for labelling of clusters in real-time using fuzzy rules based on domain/expert knowledge. We then formalise a fuzzy classificatory process that uses the dynamically labelled clusters on-board the mobile device. This model enables UDM users/applications to understand and apply the results from the clustering process in a dynamic and continuous process. We demonstrate the validity of our approach in an UDM application in the domain of road safety where we establish drunk-driving behaviour accurately and efficiently.
The paper is organised as follows. In Section 2, we review data stream clustering techniques. In Section 3, we present a conceptual overview and formalisation of our approach. Section 4 presents evaluation of our model in a road safety application. Section 5 concludes the paper.

2. Related Work

Several algorithms have been proposed for clustering data streams using sampling, creating data synopsis and approximation techniques. The following is a brief survey of each of these algorithms.
Guha et al. [9] [8] have studied clustering data streams using K-median technique. Their algorithm makes a single pass over the data and use small space. It requires O(nk) time and O(nε) space where "k" is the number of centers, "n" is the number of points and $\varepsilon<1$. The algorithm is not implemented, but the analysis of space and time requirements of it is studied analytically. They have proved that any k-median algorithm that achieves a constant factor approximation can not achieve a better run time than O(nk). The algorithm starts by clustering a calculated size sample according to the available memory into 2k, and then at a second level, the algorithm clusters the above points for a number of samples into 2k and this process is repeated to a number of levels, and finally it clusters the 2k clusters to k clusters.
Babcock et al. [3] have used exponential histogram *EH* data structure to enhance Guha et al. [8] algorithm. They have used the same algorithm described above, however the algorithm attempts to address the problem of merging clusters when the two sets off cluster centers to be merged are far apart by marinating the EH data structure. They have studied their proposed algorithm analytically.
Charikar et al [4] have proposed a k-median algorithm that overcomes the problem of increasing approximation factors in the Guha et al. [8] algorithm with the increasing in the number of levels used to result in the final solution of the divide and conquer algorithm. This technique has been studied analytically.
Domingos et al. [7] have proposed a general method for scaling up machine learning algorithms. This method depends on determining an upper bound for the learner's loss as a function in number of examples in each step of the algorithm. They have applied this method to *K-means* clustering in their algorithm Very Fast K-Means *VFKM* and decision tree classification *VFDT* techniques. These algorithms have been implemented and tested on synthetic data sets as well as real web data. The *VFKM* runs as a sequence of *K-means* executions with each run uses more examples than the previous one until a calculated statistical bound is satisfied.
O'Challaghan et al. [22] have proposed *STREAM* and *LOCALSEARCH* algorithms for high quality data stream clustering. The *STREAM* algorithm starts by determining the size of the sample and then applies the *LOCALSEARCH* algorithm if the sample size is larger than a pre-specified statistical measure. This process is repeated for each data chunk. Finally, the *LOCALSEARCH* algorithm is applied to the cluster centers generated in the previous iterations.
Aggarwal et al. [1] have proposed a framework for clustering data steams called *CluStream* algorithm. The proposed technique divides the clustering process into two components. The online component stores summary statistic about the data streams. These summarized statistics form what so called micro-clusters and the offline phase performs clustering on the generated micro-clusters according to a number of user preferences such as the time frame and the number of clusters. A number of experiments on real datasets have been conducted to prove the accuracy and efficiency of the proposed

algorithm. They [2] have recently proposed *HPStream*; a projected clustering for high dimensional data streams. *HPStream* has outperformed *CluStream* in recent results.
Keogh et al [18] have proved empirically that most cited clustering time series data streams algorithms proposed so far in the literature come out with meaningless results in subsequence clustering. They have proposed a solution approach using *K-motif* to choose the subsequences that the algorithm can work on.
In the context of ubiquitous data stream mining applications, clustering techniques have been used in VEDAS [16] – a system that monitors vehicles on the road. The focus is on outlier detection. We have developed a class of resource-aware ubiquitous data stream analysis techniques that adapt the accuracy of the results to the available computational resources on mobile and embedded devices [12]. We have developed a suite of algorithms that include a light-weight clustering algorithm (LWC), a light-weight classification algorithm (LWClass) and a light-weight frequent items analysis algorithm (LWF) [12]. In this paper we present our extensions to LWC that enable on-line labelling of clusters through fuzzy rules and subsequent fuzzy classification of ubiquitous data streams using the labelled clusters. It is noteworthy that the fuzzy rule based labelling and classification can be integrated with ubiquitous data stream clustering processes in general.

3. A Fuzzy Approach for Interpreting and Applying UDM Clustering

Online clustering, while a very valuable technique for dealing with unknown data streams, is insufficient for many UDM applications. Online classification, on the other hand, is far more suitable for applications which require time-critical decision making. However, to achieve stable classification models a considerable amount of training data is necessary. The classification models are built offline and require a considerable amount of time and resources. These are luxuries which many UDM applications do not posses.
Our model addresses the three main constraints in performing meaningful analysis of data streams on mobile devices:

1. The lack of computational resources.
2. The need for time-critical decision making.
3. The possible lack of relevant training data.

We have chosen to apply a fuzzy approach to our model. The benefits of the fuzzy approach are two-fold; they help us achieve real-time decision making, and allow a degree of uncertainty and fuzziness at the classification stage. Time-critical decision making is achieved by utilising fuzzy logic principles to express human expert knowledge, thereby facilitating the automated interpretation of clustering results. Allowing a degree of uncertainty at the classification stage makes the model more robust and flexible, and more able to handle a variety of complex data. Since the classification results are not clear-cut and absolute, a wider spectrum of possible interpretation is available, as well as possible complimentary actions.
We propose a three-stage model for intelligent data analysis of data streams on mobile devices, as shown in Figure 1 In the first stage, online clustering of incoming data streams will be performed, using an incremental clustering algorithm. Our particular contribution involves the second and third stages. In the second stage, fuzzy logic principles will be used to label the resulting clusters based on domain/expert knowledge. In the third stage of the model, the online classification stage, new incoming data streams will be assigned to the labelled clusters using notions of fuzzy degree of membership.

Figure 1: A model for applying a fuzzy logic approach to UDM

3.1 Fuzzy Labelling

The first stage of the model stage involves the online clustering of data streams. As discussed earlier, there are a number of algorithms which tackle the challenges of clustering data streams [9] [11] [15] [12]. In our model, we have chosen to use the Light Weight Clustering (LWC) algorithm, presented in

[12]. LWC is an incremental clustering algorithm, and the results of its clustering are represented as the clustering centres. Once we have attained the clustering results, the next step is to label them.
In the second stage of our model we analyse the attained clustering results in conjunction with available expert/domain knowledge in the appropriate field, in order to interpret and annotate these results. Once we interpret the clustering results and label them, we can use these for online classification. Fuzzy logic systems often combine numerical or computational information with linguistic information, where linguistic information usually represents expert knowledge. In [21] fuzzy logic systems are described as systems which are capable of combining "objective knowledge" and "subjective knowledge". In the case of our system, the "objective knowledge", or numerical information, is the clustering models built at the online clustering stage. The "subjective knowledge", or linguistic information, on the other hand, is the expert/domain knowledge available in the field in which the UDM application is developed.
We use the expert knowledge to label the clustering results by constructing a knowledge base, as shown in Figure 2. The knowledge base is composed of rules which represent the expert/domain knowledge, or information. As long as the expert knowledge can be represented as rules it is viable to our model. The rules must relate to the particular attributes according to which the data streams are clustered, as well as to the final categories by which we later classify new, unseen, data items.

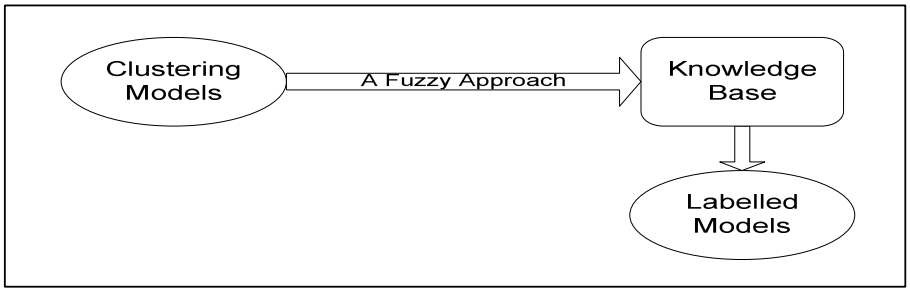

Figure 2: Clustering to classification through fuzzy logic

The attributes by which a clustering is done are always known beforehand, even if the types of classes themselves are not. In the case of a random clustering of a group of people, for example, the attributes for clustering can be *height*, *weight*, *age* and *sex*, or they could be *salary*, *number of rooms in the house* and *number of children*. In the first example, we might want to know if people are likely to suffer from high blood pressure, in which case a pertinent rule might be "*the higher the weight, in relation to the height, the higher the likelihood of high blood pressure*", or "*the higher the age, the higher the likelihood of high blood pressure*" and so on for the rest of the attributes. We can then label the cluster which contains people who are older and have a higher weight-height ratio as those most likely to suffer from high blood pressure, and the cluster containing younger people with a lower weight-height ratio as the least likely to suffer from high blood pressure. The rest of the clusters in between can be rated on a curve. The rules relevant for the second example would be completely different and depend on the categories by which we want to classify the group of people, for instance, their socioeconomic status.

3.2 Online Classification

In the third stage of the model online classification will be performed, using the labelled clusters constructed at the in the previous stage. To achieve a more robust model, able to handle uncertainties and "fuzziness" in the data, we employ a fuzzy logic approach at the online classification stage.
One of the characteristics of fuzzy logic, and in particular fuzzy sets, is that the question of membership to a set is not answered with a definitive *yes* or *no* but rather a degree of membership. This trait enables applications to have greater flexibility and allows a far wider range of actions based on the varying degrees of membership. For these reasons we have chosen to incorporate the notion of a degree of membership to our own model. The online classification of new incoming data items will not merely classify the data item to one class, but rather show the degree of probability of that data item belonging to each of the known classes.
When classifying an incoming data stream item and assigning it to one of the available classes the following factors are considered:

1. The distance of the new data item from the class centre.
2. The weight, or size, of the class.

3. The distance of the data item from the other class centres.

It is important to note that when classifying a data item its relation to *each* of the classes is taken into consideration. The results of classification based on these factors will therefore include a degree of probability of the data item belonging to each of the classes.

3.3 Formalization of the Algorithms

In this section we present the formalization of the algorithms used in the cluster labelling stage and in the online classification stage.

3.3.1 Formalization of Fuzzy Cluster Labelling

For the purpose of our algorithm for cluster labelling, we have assumed that the knowledge base includes at least one expert rule regarding each clustering attribute. We have also assumed that each such rule would allow us to sort the clusters according to the particular attribute, or attributes, it pertains to. This algorithm is relevant in cases where the characteristic by which the labelling is done, and the eventual classification, can be categorised by varying degrees of strength, from the least to the most.

The rules should be in the form of "IF - THEN" rules, where the antecedent of the rule represents the way an attribute, or attributes, affect the characteristic by which we classify. For instance, let us say we want to classify a number of data items according to the characteristic X, using the attributes A B and C. We may know that as the attribute A increases, so does X. This fact is represented using the following notation:

IF A^i => X^i

We may also know that as the attribute A increases and attribute B decreases, the characteristic X increases in strength. A way to represent this knowledge as expert/domain rules which can be used in our algorithm would be:

IF A^i and B^d => X^i

Where A^i stands for increases in attribute A and A^d stands for decreases in attribute A.

Figure 3 shows the steps of the fuzzy cluster labelling algorithm as follows:

1. Initialization and declaration of variables.
2. The *LWC* function performs the LWC incremental clustering algorithm and returns a set of weighted cluster centres $C = \{c_1, c_2, \ldots, c_n\}$.
3. The cluster centres are sorted according to each rule. The position of the cluster centre in each sort is maintained in a set $S(c_i) = \{s_1, s_2, \ldots, s_m\}$ where s_i represents the position of the cluster c_i in the i^{th} sort.
4. The total position of the cluster centre $P(c_i)$ is determined using $S(c_i)$ as follows: $P(c_i) = \sum_{i=1}^{numberofrules} S(ci)$

   To illustrate steps 3 and 4 we use the example rules from the above discussion (i.e. IF A^i => X^i and IF A^i and B^d => X^i). Let us add another rule:

   IF C^d => X^i.

   Now let us sort three cluster centres (CC1, CC2 and CC3) according to each rule:

Table 1: Cluster centres sorted by rules

Position	Rule 1	Rule 2	Rule 3
1	CC3	CC3	CC3
2	CC1	CC2	CC1
3	CC2	CC1	CC2

   We can now sum up the positions for each cluster centre, which is its overall Position:

   CC1 = 2+3+2 = 7

   CC2 = 3+2+3 = 8

   CC3 = 1+1+1 = 3

5. The cluster centres are sorted by their total position, $P(c_i)$, from the lowest to the highest, or from the least to the most. Their labelling is according to the positions after the sort.

```
1. Initialization/Declaration:
       a. Let LWC be a function performing the LWC clustering algorithm.
       b. Let R be a set of rules representing domain/expert knowledge
          Let m be the number of rules: R = {r1, r2, …, rm}
          Let ri represent the ith rule
       c. Let C be a set of cluster centres
          Let n be the number of cluster centres: C = {c1, c2, …, cn}
          Let ci represent the ith cluster centre:
             i. Let S(ci) be a set representing the position of the cluster
                centre in each sort by a rule:
                S(ci) = {s1, s 2, …, s m}
                Let si represent the position of the cluster centre in the ith
                sort (by the ith rule)
            ii. Let P(ci) be the overall position of the cluster centre
                after sorting according to the rules
2.  C = LWC()
3.  For i = 1 to m
       Sort cluster centres by each rule: Sort(C, ri)
       For j = 1 to n
              si(cj)= j
       Until Done
    Until Done
4.  For i = 1 to n
       For j = 1 to m
             P(ci) = P(ci) + sj(ci)
       Until Done
    Until Done
5.  Sort cluster centres by overall position: Sort(C, P)
```

Figure 3: Fuzzy cluster labelling algorithm

The Algorithm presented above has a certain limitation in that it is to be confined to IF – THEN rules where the antecedent of the rule has either a direct or indirect proportional relation to the consequent, as shown in Figure 4. However, as the model develops, and more accurate real data is obtained, the rules in the knowledge base can be refined over time to be able to represent the type of information shown in Figure 5.

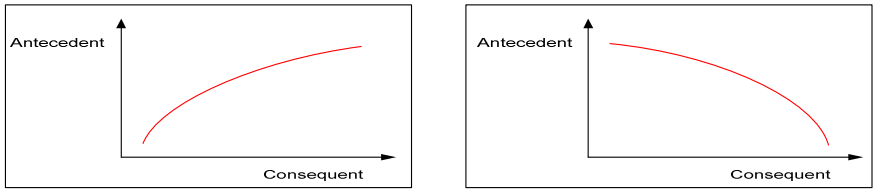

Figure 4: A representation of simple direct or indirect relationship

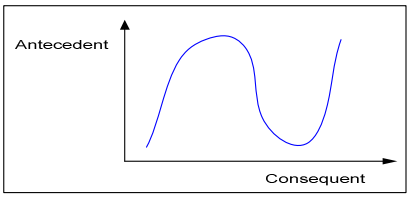

Figure 5: A representation of more complex information

3.3.2 Formalization of Fuzzy Classification Algorithm

Taking into consideration the three factors enumerated above: the distance of the data item from the cluster centre, the cluster's size, and the distance of the data item from the other cluster centres, we have constructed a fuzzy classification algorithm for classifying incoming data items which includes a data item's degree of membership in each of the known labelled clusters.

As we stated, in the first stage we use the LWC incremental clustering algorithm for performing online clustering. The outcome of the LWC algorithm are the weighted cluster centres of the resulting

clusters, where the weight of the cluster centre represents the size of the cluster. The cluster centres are then labelled according to expert/domain knowledge rules, and are, in effect, the classes to which we assign new data items. Our approach for calculating the distance of an item from each cluster and its degree of membership for each cluster is as follows:

1. For each of the labelled clusters:

   Calculate the distance of the data item to the cluster centre. Merely calculating the distance of a data item from a cluster centre is insufficient, as we need to take into consideration the cluster's size as well, or the number of items in the cluster, which is represented by the cluster's *weight*. The weight can be perceived as a cluster's gravitational pool on the data item. Taking it into consideration greatly reduces the affect of anomalies such as outliers. By multiplying the inverse distance to the cluster centre by the weight of the cluster, and dividing by the combined weight of all the cluster centres, we get an accurate and representative distance, or affiliation, of the data item to the particular cluster:

$$C_x = \frac{1}{dist} \times \frac{weight}{\frac{numberofcenters}{\sum_{i=1} weight}} \qquad (1)$$

```
1.  Initialization/Declaration:
        a.  Let C be a set of cluster centres
            Let m be the number of cluster centres: C = {c1, c2, …, cm}
            Let ci represent the ith cluster centre:
                i.  Let A(ci) be a set of attributes
                    Let n be the number of attributes: A(ci) = {a1, a2, …, an}
                    Let ai represent the ith attribute of the cluster centre
               ii.  Let W(ci) be the weight of the cluster centre (representing
                    the size of the cluster)
        b.  Let DI be a data item (note – each data item has the same attributes
            as those represented in the cluster centre)
        c.  Let WS be the total weight of all the cluster centres
        d.  Let D be the total distances between each of the data item's
            attributes and the cluster centre's attributes
        e.  Let DW be a set representing the distance between a data item and
            each cluster centre, taking the cluster centre's weight into account
            Let dwi represent the distance/weight of a data item to the ith
            cluster centre
        f.  Let DWS be the total distance/weight between the data item and the
            cluster centres
        g.  Let M be a set representing the degree of membership of a data item
            to each cluster centre
            Let mi represent the degrees of membership of a data item to the ith
            cluster centre
2.  For i = 1 to m
        WS = WS + W(ci)
    Until Done
3.  For i = 1 to m
        D = 0
        For j = 1 to n
            D = D + ( aj(DI) – aj(ci) )
        Until Done
        dwi = (1 / D) * (W(ci) / WS)
    Until Done
4.  For i = 1 to m
        DWS = DWS + dwi
    Until Done
5.  For i = 1 to m
        mi = (DW / DWS) * 100
    Until Done
```

Figure 6: Fuzzy classification algorithm

2. In order to get the probability of the data sample belonging to each of the clusters, we need to consider the distance of the data item to the cluster centre, in relation to its distance to the other cluster centres. We therefore divide the result of the first step by the combined results of the first step of all the cluster centres, to get the degree of membership of the data item to the labelled cluster:

$$cDegree\ of\ membership = \frac{Cx}{\sum_{i=1}^{numberofcenters} Ci} \quad (2)$$

The fuzzy classification algorithm formalised in Figure 6 is as follows:

1. Initialization and declaration of variables.
2. The total weight of all the cluster centres is represented as *WS*. As the weight of each cluster centre is the number of items assigned to it, the combined weight of all the cluster centres actually represents the total number of data items clustered.
3. For each cluster centre, the overall, representative, distance from it to the new data item, $DW(c_i)$, is calculated by combining the distances between each of the attributes, taking into account the weight of the cluster centre. Distance is calculated using (1).
4. The total distance of the new data item from all the cluster centres is represented as
$$DWS = \sum_{i=1}^{numberofclusters} DW(\text{ci}) \quad (3)$$
5. The percentile degree of membership of the new data item to each labelled cluster is calculated by dividing the data item's distance from the cluster centre, $DW(c_i)$, by the total distance, *DWS*. The Degree of membership is calculated using (2).

4. Evaluation of the Model in a Road Safety Scenario

As we discussed, we have created a model for using UDM techniques, in combination with fuzzy logic principles. We demonstrate the application and evaluation of this model in the field of road safety. In this section we will present the results of applying the labelled clusters based on fuzzy rules to facilitate online classification of data streams, in order to detect, and alert to, dangerous drunk driving behaviour. Our goal is to be able to make sense of previously unseen data. Rather then merely applying unsupervised data mining techniques to a new data sample, we use a fuzzy logic approach which enables us to classify the unseen data sample online, in real-time.

For the purpose of evaluation, our algorithm is used in the case of drunk driving behaviour. The first step of the algorithm involves randomly generating data streams which emulate the kind of sensory data that can be obtained in a moving vehicle. Incremental clustering is then performed on the unseen data, on the on-board device, using the LWC algorithm [12], which is a lightweight one-look incremental clustering algorithm designed for operating in real time on resource constrained devices.

The generated data is taken from a study presented in [20], where participants used a driving simulator with deferent degrees of blood alcohol concentration. The following response measures were recorded in the study:

- Reaction time to peripheral signals (sec).
- Correct responses to peripheral signals (number).
- Speed deviation (mph).
- Lane position deviation (ft).
- Collisions (number).
- Times over speed limit (number).

In the second step of the algorithm, we transmit the clustering results to the server, where, using fuzzy logic, we combine these results with expert knowledge. This gives us labelled classes of drunk driving behaviour, ordered by the degree of drunkenness; from the least drunk to the most drunk. The third step of the algorithm is the online classification. The labelled classification models are sent back to the on-board device, where they are used to classify new, unknown, data samples.

As we discussed, in the first step of the algorithm we cluster generated data streams. The data is generated in such a fashion that every five minutes the behaviour of the driver is randomly changed. Using the data recorded in the [20] study, we generate four types of drunk driving behaviour; sober, borderline, drunk and very drunk, as shown in Figure 7. For each type, we randomly generate a value from a pre-defined range, obtained from the study, for each of the attributes mentioned above.

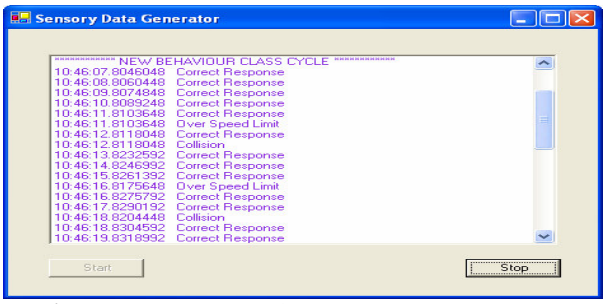

Figure 7: Screen shot of the Sensory Data Generator

The data is then clustered on-board the device, as shown in Figure 8. After some testing, we have defined a threshold to the LWC algorithm which gives us optimal clustering results. The weight of a cluster centre represents the number of items in that cluster. The testing of the on-board device was done on iPAQ with 64 MB, running Microsoft Windows CE version 3.0.9348 with StrongArm processor as shown in Fig 5. The program has been developed using Microsoft embedded Visual C++ 4.0.

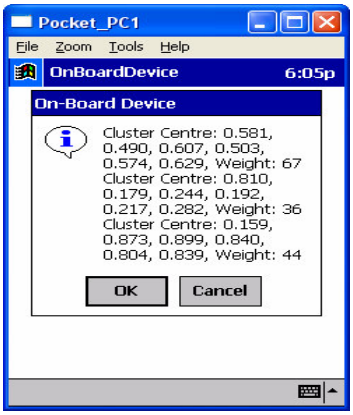

Figure 8: Screen shot of clustering results (using LWC)

In the second step of the algorithm, the clustering centres are sent to the central server where, using expert knowledge, they are ordered by degree of drunkenness. The expert knowledge includes rules such as:

- The *higher* the number of "Correct responses to peripheral signals", the *less* drunk the driver.
- The *higher* the number of "Times over speed limit" ", the *more* drunk the driver.

Figure 9 shows the labelling results.

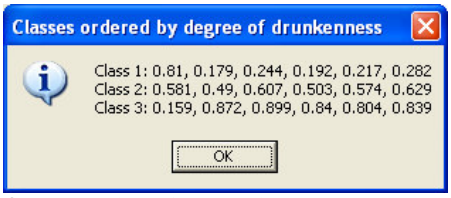

Figure 9: Screen shot of classes labelled by degree of drunkenness

In the third stage of the algorithm the classification models are used on-board the device to classify new data items. Here again we apply the fuzzy logic approach. Instead of merely classifying a driver's behaviour to one of the available classes, we find its degree of membership, or the probability it of belonging, to each of the classes. Figure 10 shows the classification results of two different, previously unseen, data samples.

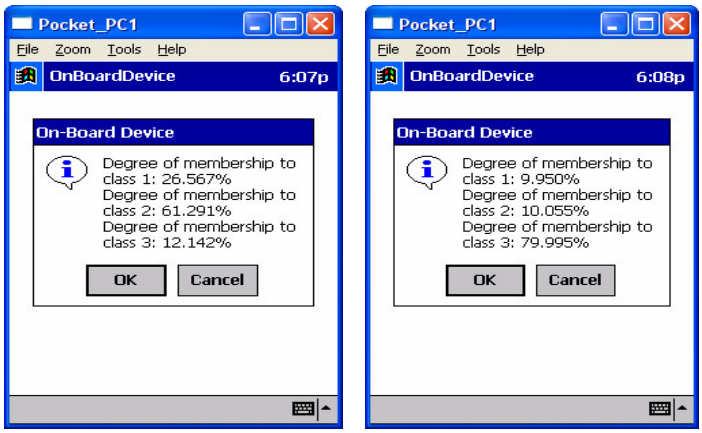

Figure 10: Screen shots of online classification results (samples 4 and 5 in Table 1)

Table 2 shows the results of the classification of randomly generated new data samples. Each of the data sample was generated from one of the four original classes of driving behaviour mentioned above (Observed Behaviour). The classifying application is, of course, unaware of these classes, but it allows us to test our expected results.

Table 2: Classification of different unseen data samples.

	Generated Attributes						Observed Behaviour	Expected Result		
	Correct responses	Collisions	Times over speed limit	Reaction time	Speed deviation	Lane deviation		Class1 (least drunk)	Class2	Class3 (most drunk)
1	58	4	4	2.84	3.09	1.57	Sober	79.4 %	12.5 %	8.1%
2	54	8	10	2.93	3.42	1.9	Drunk	21.5 %	52.2 %	26.3 %
3	49	10	12	3.08	3.54	1.98	Very Drunk	9.2%	9.5%	81.3 %
4	56	6	9	2.87	3.38	1.78	Borderline	26.6 %	61.3 %	12.1 %
5	48	9	13	3.05	3.6	2.01	Very Drunk	9.9%	10.1 %	80%
6	54	7	10	2.94	3.5	1.85	Drunk	20.8 %	55.8 %	23.4 %

We can make several observations from the results of the algorithm. Firstly, let us consider the results of the first stage and second stage of the algorithm. After the online clustering and offline labelling we ended up with three classes of driving behaviour with varying degrees of drunkenness, from least drunk to most drunk, even though we originally generated the data as four distinct driving behaviour classes. This is because the difference between the borderline and the drunk driving behaviour is difficult to distinguish, while the distinction between these two classes and the sober and very drunk behaviour classes is very clear.

How then do we know which driver is borderline drunk and which driver certainly is drunk? This is helped by the results of the third stage of the algorithm. Because we have the degree of membership to each class of behaviour, if a new data sample has the highest degree of membership to class number 2 (which includes borderline and drunk driving), we can determine the level of drunkenness by observing to which of the other to classes that sample has a higher degree of membership. We can see, for instance, that both the second and fourth test samples in Table 2 have the highest probability of belonging to class number 2. However, for the fourth sample, the second highest probability is of belonging to class number 1 (sober), while the second sample has the second highest probability of belonging to class number 3 (very drunk). This tells us that the driver from the second sample is

drunker than the driver in the fourth sample. Appropriate action can now be taken; for the less drunk driver, a local warning might suffice, while for the driver who displays the more serious attributes of drunk driving, a more general alarm, perhaps to other cars in the vicinity, is required.
It is evident that our method of combining UDM techniques with a fuzzy logic approach provides results which are easily comprehensible, and which allow for an immediate, as well as appropriate, respond, without the need of human intervention. While other similar applications require human experts to evaluate results and perhaps recommend a response, we include human knowledge and expertise as part of the application, thereby, "cutting the middleman".

5. Conclusions

In this paper, we have proposed a model which enables the transition from the clustering models obtained from mining data streams to interpretable and applicable classificatory models, through a continuous, online operational process. Our model addresses the three main constraints in performing meaningful analysis of data streams on mobile devices:

- The lack of computational resources.
- The need for time-critical decision making.
- The possible lack of relevant training data.

In our model, we have employed a fuzzy approach to address the issue of dynamic interpretation and classification of data. The benefits of our fuzzy approach are two-fold:

- The incorporation expert/domain knowledge to the data analysis process helps us achieve time-critical decision making.
- Allowing a degree of uncertainty and fuzziness at the classification stage helps us achieve more flexible and robust models, which permit a greater scope of action.

In order to illustrate the feasibility and demonstrate the validity of our proposed model we have applied it to the field of road safety. In particular, we have employed our three-stage model for fuzzy data analysis to the task of monitoring dangerous driving behaviour on-board vehicles, using mobile devices. Our model for performing intelligent data analysis on-board resource constrained devices in real time is particularly suited to the task of monitoring dangerous driving behaviour, as it enables online mining of sensory data, dynamic result interpretation and real time decision making, which are crucial to accident prevention.
For the purpose of evaluation, our implementation concentrated on the case of drunk driving behaviour. Using our prototype implementation we have demonstrated the accuracy and sensitivity of our model for performing online classification using dynamically labelled clustering results. Our prototype produced results which are easily comprehensible, and which allow for an immediate and appropriate respond, without the need of human intervention.

References

1. C. Aggarwal, J. Han, J. Wang, P. S. Yu, A Framework for Clustering Evolving Data Streams, Proc. 2003 Int. Conf. on Very Large Data Bases (VLDB'03), Berlin, Germany, Sept. 2003.
2. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, A Framework for Projected Clustering of High Dimensional Data Streams, Proc. 2004 Int. Conf. on Very Large Data Bases (VLDB'04), Toronto, Canada, Aug. 2004.
3. B. Babcock, M. Datar, R. Motwani, L. O'Callaghan: Maintaining Variance and k-Medians over Data Stream Windows, to appear in Proceedings of the 22nd Symposium on Principles of Database Systems (PODS 2003).
4. M. Charikar, L. O'Callaghan, and R. Panigrahy. Better streaming algorithms for clustering problems In Proc. of 35th ACM Symposium on Theory of Computing (STOC), 2003.
5. Chong, S,K., Loke, S, W., and Krishnaswamy, S., (2005), Wireless Sensor Networks: From Data to Context to Energy Saving, Accepted for publication in the Proceedings of the International Workshop on Ubiquitous Data Management (UDM 2005) to be held in conjunction with the 21st International Conference on Data Engineering (ICDE 2005), Tokyo, Japan, April 4, IEEE Press.
6. Chen, R., Sivakumar, K., Kargupta, H., An Approach to Online Bayesian Learning from Multiple Data Streams, Workshop on Ubiquitous Data Mining for Mobile and Distributed Environments, Held in Conjunction with Joint 12th European Conference on Machine Learning (ECML'01) and 5th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'01), Freiburg, Germany, September 3-7, 2001.

7. P. Domingos and G. Hulten, A General Method for Scaling Up Machine Learning Algorithms and its Application to Clustering, Proceedings of the Eighteenth International Conference on Machine Learning, 2001, 106--113, Williamstown, MA, Morgan Kaufmann.
8. Sudipto Guha, Adam Meyerson, Nina Mishra, Rajeev Motwani, and Liadan O'Callaghan, Clustering Data Streams: Theory and Practice TKDE special issue on clustering, vol. 15, 2003.
9. Guha, S., Mishra, N., Motwani, R., and O'Callaghan, L., Clustering data streams, in Proc. FOCS, p. 359–366, 2000.
10. Grossman, R., Supporting the Data Mining Process with Next Generation Data Mining Systems, Enterprise Systems, August 1998.
11. Gupta, C., and Grossman, R. L., GenIc: A Single Pass Generalized Incremental Algorithm for Clustering, International Conference on Data Mining, SIAM 2004.
12. Gaber, M. M., Krishnaswamy, S., Zaslavsky, A., Cost-Efficient Mining Techniques for Data Streams, Australasian Workshop on Data Mining and Web Intelligence (DMWI2004), Dunedin, New Zealand, 2004.
13. Gaber, M, M., Zaslavsky, A., and Krishnaswamy, S., (2004), Towards an Adaptive Approach for Mining Data Streams in Resource Constrained Environments, Accepted for publication in the Proceedings of Sixth International Conference on Data Warehousing and Knowledge Discovery - Industry Track (DaWak 2004), Zaragoza, Spain, 30 August - 3 September, Lecture Notes in Computer Science (LNCS), Springer Verlag.
14. Gaber, M, M., Krishnaswamy, S., and Zaslavsky, A., (2005), Resource-Aware Mining of Data Streams, Accepted for publication in *Journal of Universal Computer Science - Special Issue on Knowledge Discovery in Data Streams*, edited by Jesus S. Aguilar-Ruiz and Joao Gama, August 2005.
15. Gaber, M, M., Zaslavsky, A., and Krishnaswamy, S., (2005), Mining Data Streams: A Review, Accepted for publication in *ACM SIGMOD Record,* Vol. 34, No. 2, June 2005, ISSN: 0163-5808.
16. H. Kargupta, R. Bhargava, K. Liu, M. Powers, P. Blair, S. Bushra, J. Dull, K. Sarkar, M. Klein, M. Vasa, and D. Handy., (2004), "*VEDAS: A Mobile and Distributed Data Stream Mining System for Real-Time Vehicle Monitoring*." Proceedings of the SIAM International Data Mining Conference, Orlando.
17. Krishnaswamy S., Loke S. W., Rakotonirainy A., Horovitz O., and Gaber M. M. (2005), *"Towards Situation-awareness and Ubiquitous Data Mining for Road Safety: Rationale and Architecture for a Compelling Application"*, Proceedings of Conference on Intelligent Vehicles and Road Infrastructure, Melbourne, Australia 16-17 February.
18. E. Keogh, J. Lin, and W. Truppel. Clustering of Time Series Subsequences is Meaningless: Implications for Past and Future Research. In proceedings of the 3rd IEEE International Conference on Data Mining. Melbourne, FL. Nov 19-22, 2003.
19. Kargupta, H., Park, B., Pittie, S., Liu, L., Kushraj, D., and Sarkar, K. MobiMine: Monitoring the Stock Market from a PDA. ACM SIGKDD Explorations, Volume 3, Issue 2. Pages 37--46. ACM Press, January 2002.
20. Moskowitz, H., Burns, M., Fiorentino, D., Smiley, A., Zador, P., Driver Characteristics and Impairment at Various BACs, Southern California Research Institute, August 2000.
21. Mendel, J. M., Fuzzy logic systems for engineering: A tutorial, Proceedings of the IEEE, 83(3):345–377, March 1995.
22. L. O'Callaghan, N. Mishra, A. Meyerson, S. Guha, and R. Motwani. Streaming-data algorithms for high-quality clustering. Proceedings of IEEE International Conference on Data Engineering, March 2002.
23. S. Tanner, M. Alshayeb, E. Criswell, M. Iyer, A. McDowell, M. McEniry, K. Regner (2002), *"EVE: On-Board Process Planning and Execution"*, Earth Science Technology Conference, Pasadena, June, CA, USA.
24. Zaki, M, J., (2002), "*Editorial: Online, Interactive and Anytime Data Mining*", SIGKDD Explorations, Vol. 3, Issue 2, January. Available Online: http://www.acm.org/sigs/sigkdd/explorations/issue3-2/contents.htm #Editorial